

Observational Report LAB 9: MD5 Collision Attack

ACS 545| Cryptography and Network Security

April 3, 2024

Ashutosh Mishra

Professor: ZesHeng Chen

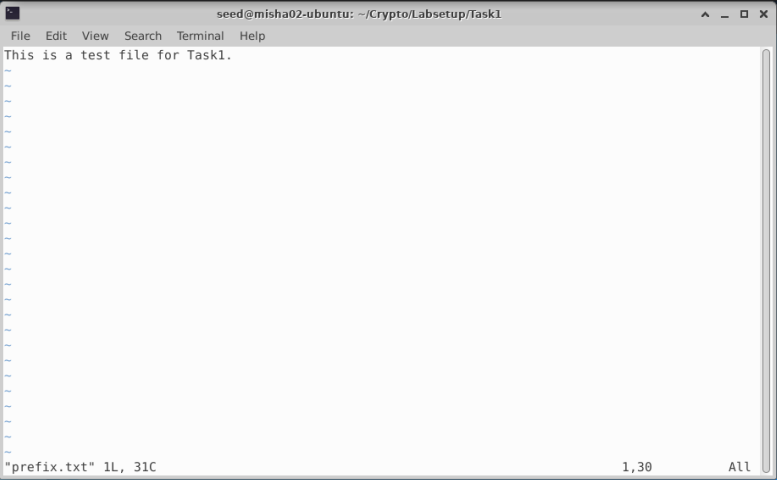
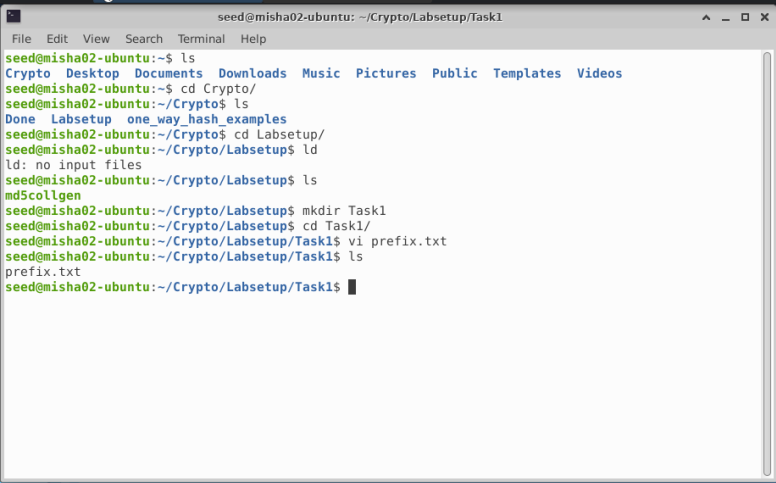
* Observation Criteria: -

Note: -

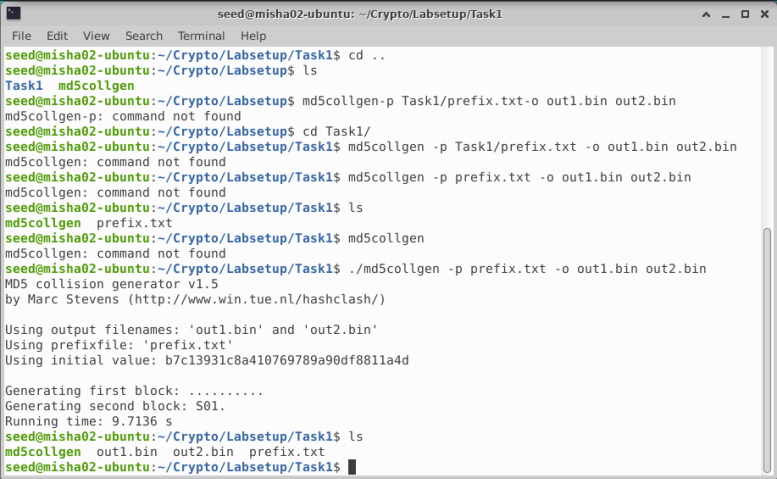
1. Used Ubuntu Seed (v20.04 Focal) on Google Cloud Platform
2. Used Real VNC Viewer to perform the actions on GUI
3. All actions are being performed under username – **seed**

* Task 1 (Generating Two Different Files with the Same MD5 Hash): -

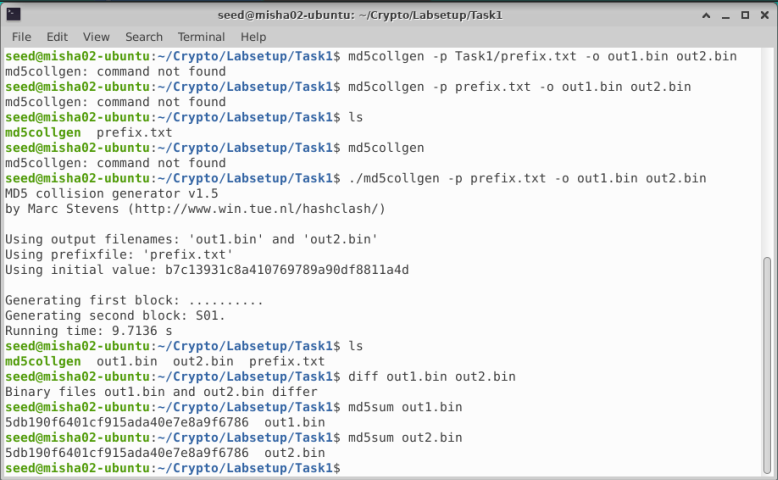
For this task, we need to first create a file prefix.txt.



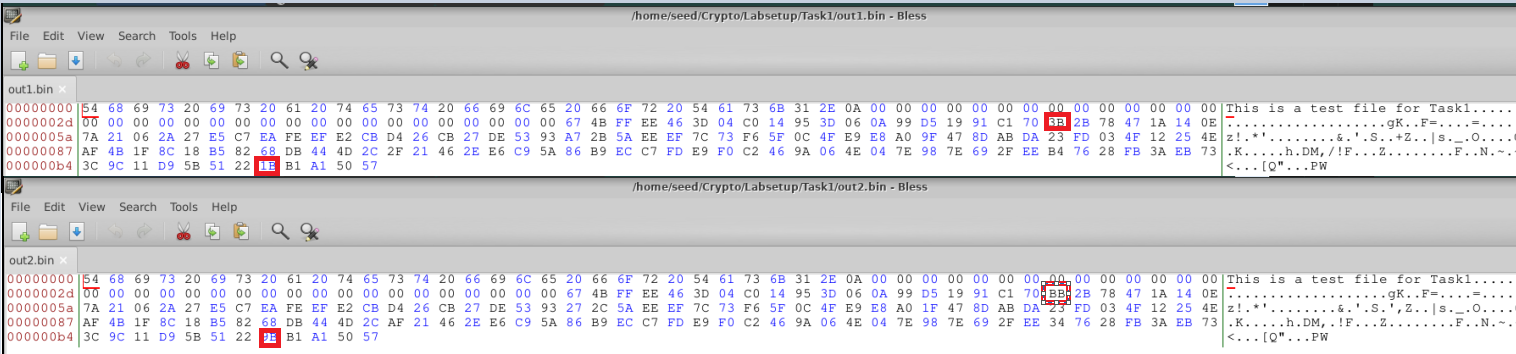
Now we will use the md5collgen to generate two output files **out1.bin** and **out2.bin**.



Now let’s check if the output is distinct or not by using diff command and comparing the md5sum for both the files. As we can see that the output files are differing. But, the md5 values remain identical.



Let’s view and compare the files using bless. We can observe that the files look almost the same except for some of the entries differ slightly.



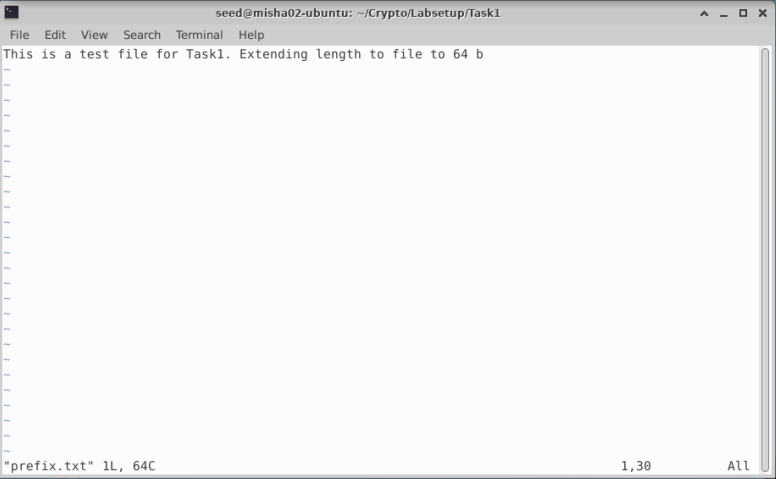
Question 1: - If the length of your prefix file is not multiple of 64, what is going to happen?

Ans: - If the length of the prefix is not in multiple of 64, then zeroes will be padded.

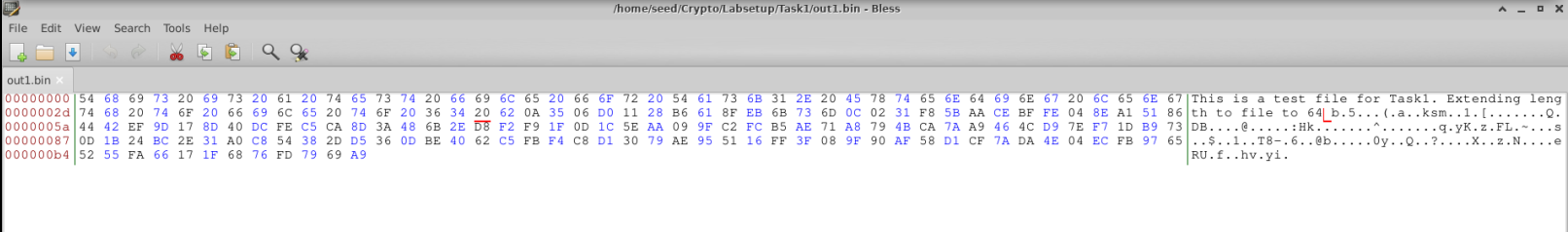
Question 2: - Create a prefix file with exactly 64 bytes, and run the collision tool again, and see what happens.

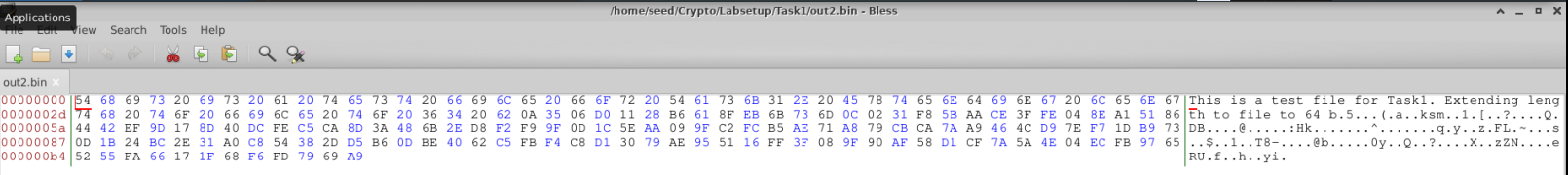
Ans: - Creating a 64 byte file.





Running the collision tool again and comparing outputs.





In both the cases, there is no zero-padding observed. In first case, there was some zero-padding

done.

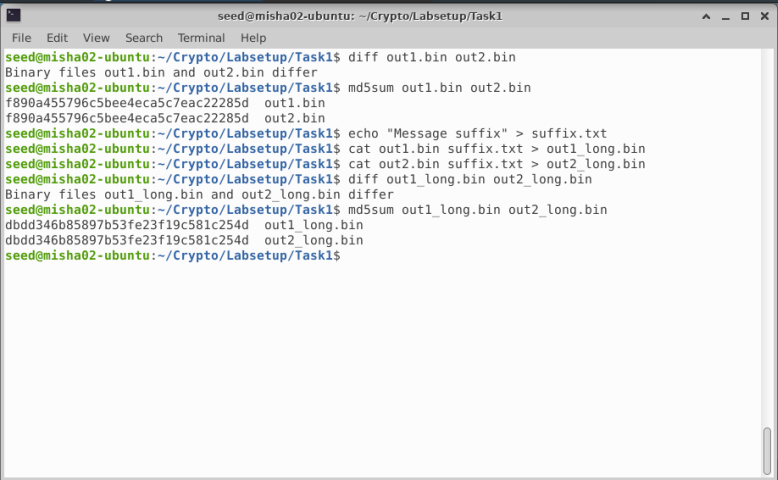
Question 3: - Are the data (128 bytes) generated by md5collgen completely different for the two output files? Please identify all the bytes that are different.

Ans: - Yes, the data generated by md5collgen are different. There were 4 bits that were different for Case2 where input length was 64 bytes. The bytes that were different are highlighted below.



* Task 2 (Understanding MD5’s Property): -

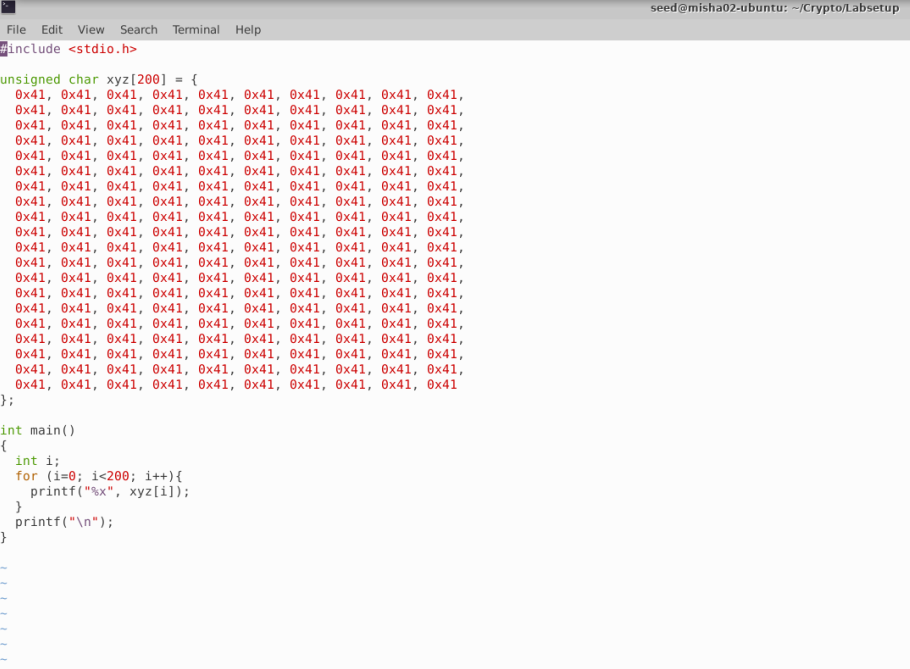
In this task, we will be performing the following steps.



As we can see, even after adding the same text i.e. suffix.txt to both the binary files keeps the hash value same however, the binary files are differing.

* Task3 (Generating Two Executable Files with the Same MD5 Hash): -

In this task, we will be using the following C program.



Let’s compile the code and check the output.

A screenshot of a computer

Description automatically generated

Let’s open the a.out file and find location 3040. The decimal equivalent of this place is 12352. Now, let’s create the prefix file.

A screenshot of a computer

Description automatically generated

After this let’s create two out files using md5collgen.

A screenshot of a computer

Description automatically generated

Let’s create the suffix file now.

A screenshot of a computer

Description automatically generated

Creating P & Q files.

A screenshot of a computer

Description automatically generated

Creating a1.out and a2.out by concatenating the files.

A screenshot of a computer

Description automatically generated

Let’s check if both the files differ. And as per the screenshot below, we can confirm that both files are differing.

A screenshot of a computer

Description automatically generated

Now, let’s compare the md5 values for both the files.

A screenshot of a computer

Description automatically generated

We can see that the md5 values are identical. Let’s run both the files and compare the outputs if they are different.

A screenshot of a computer

Description automatically generated

As we can see that although both appear similar, there are minor differences as highlighted.

We successfully generated two different executable files with same MD5 values.

* Task 4 (Making the Two Programs Behave Differently): -

For this task I have used the following code.

A screenshot of a computer

Description automatically generated

Let’s compile and execute the code.

A screenshot of a computer

Description automatically generated

Creating the prefix file at 12352.

A screenshot of a computer

Description automatically generated

Creating the out files with the same MD5 but different P & Q values.

A screenshot of a computer

Description automatically generated

Creating the suffix file.

A screenshot of a computer

Description automatically generated

Now, let’s create P, Q, suffix1 and suffix2 files.

A screenshot of a computer

Description automatically generated

Now we will generate a1.out and a2.out files.

A screenshot of a computer

Description automatically generated

Let’s run both the scripts.

A screenshot of a computer

Description automatically generated

We can observe that like the previous task, MD5 for both the files are identical, however the content is different. Also, a change that now both files behave differently.

Hence, the task is successfully completed.

* Conclusion: -

The overall experience was exciting. I will be going deeper into the subject and try to discover new ways to handle how we had practiced in lab. Having a base knowledge of how MD5 Collision attack works, I would like to do a deep dive into this subject to find out more about how to protect the target machines from attacks that occur in a simple manner. One immediate way to think is by appending a checksum like hash value to the script to know if it was changed.